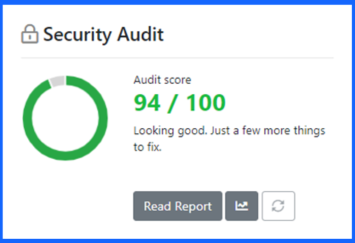
SkyNET.

SECURITY AUDIT REPORT FOR

FTX Classic

FTTC.sol


Security Audit

Audit score

94 / 100

Looking good. Just a few more things to fix.

[illegible]

CERTIFIED

Confidential

FTTC.sol / Serial no. 1827101900012019 / 11.11.2022

01. Audit

This document includes the results of the audit performed by the SkyNet auditors team on the FTX Classic project.

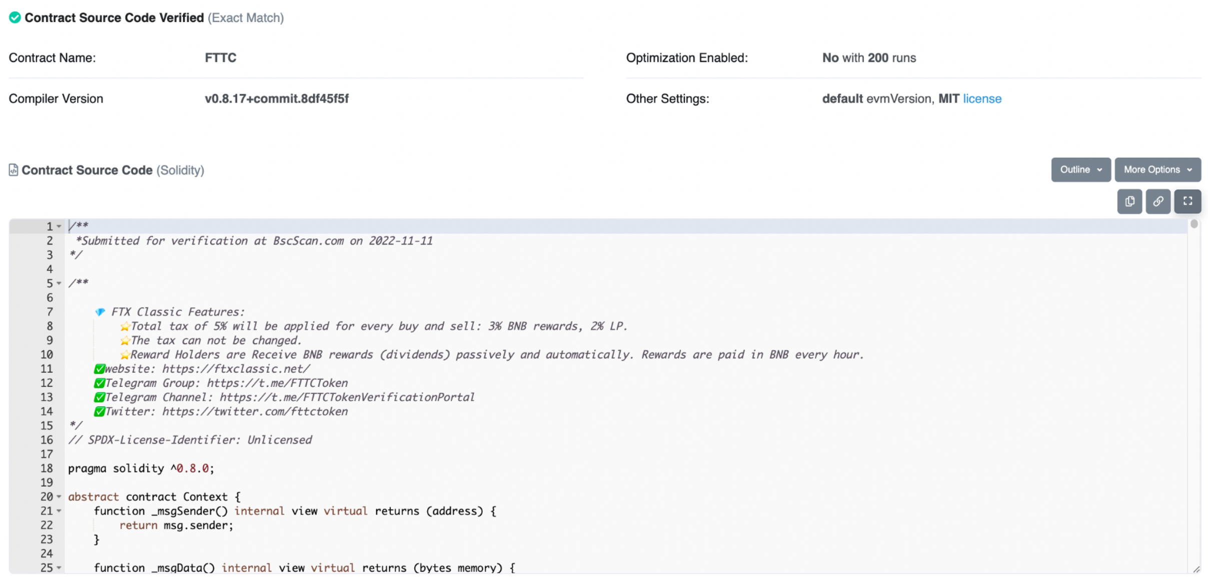
Audit Start Time:
November 11, 2022

Audit End Time:
November 11, 2022

Audited Source File's

Binance Smart Chain Address:

FTTC: 0xbe5fcd51dbb5080330c241665200e7a283e6f5c7



Contract Source Code Verified (Exact Match)

Contract Name:	FTTC	Optimization Enabled:	No with 200 runs
Compiler Version	v0.8.17+commit.8df45f5f	Other Settings:	default evmVersion, MIT license

Contract Source Code (Solidity)

```
/**
 *Submitted for verification at BscScan.com on 2022-11-11
*/

/**

    💎 FTX Classic Features:
       ⭐Total tax of 5% will be applied for every buy and sell: 3% BNB rewards, 2% LP.
       ⭐The tax can not be changed.
       ⭐Reward Holders are Receive BNB rewards (dividends) passively and automatically. Rewards are paid in BNB every hour.
    ✅website: https://ftxclassic.net/
    ✅Telegram Group: https://t.me/FTTCToken
    ✅Telegram Channel: https://t.me/FTTCTokenVerificationPortal
    ✅Twitter: https://twitter.com/fttctoken
*/
// SPDX-License-Identifier: Unlicensed

pragma solidity ^0.8.0;

abstract contract Context {
    function _msgSender() internal view virtual returns (address) {
        return msg.sender;
    }

    function _msgData() internal view virtual returns (bytes memory) {
```

https://bscscan.com/address/0xbe5fcd51dbb5080330c241665200e7a283e6f5c7#code

The goal of this audit is to review FTTC solidity implementation for its swap functions, study potential security vulnerabilities, its general design and architecture, and uncover bugs that could compromise the software in production.

We make observations on specific areas of the code that present concrete problems, as well as general observations that traverse the entire codebase horizontally, which could improve its quality as a whole.

This audit only applies to the specified code, software or any materials supplied by the FTX Classic team for specific versions.
Whenever the code, software, materials, settings, environment etc is changed, the comments of this audit will no longer apply.

— Disclaimer

Note that as of the date of publishing, the contents of this report reflect the current understanding of known security patterns and state of the art regarding system security. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk.

The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. If the audited source files are smart contract files, risks or issues introduced by using data feeds from offchain sources are not extended by this review either.

Given the size of the project, the findings detailed here are not to be considered exhaustive, and further testing and audit is recommended after the issues covered are fixed.

To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services.

FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

— Methodology

The above files' code was studied in detail in order to acquire a clear impression of how its specifications were implemented. The codebase was then subject to deep analysis and scrutiny, resulting in a series of observations. The problems and their potential solutions are discussed in this document and, whenever possible, we identify common sources for such problems and comment on them as well.

The SkyNet auditing process follows a routine series of steps:

1. Code review that includes the following

i. Review of the specifications, sources, and instructions provided to SkyNet to make sure we understand the size, scope, and functionality of the project's source code.

ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.

iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to SkyNet describe.

2. Testing and automated analysis that includes the following:

i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run the test cases.
ii. Symbolic execution, which is analysing a program to determine what inputs cause each part of a program to execute.

3. Best practices review, which is a review of the source code to improve maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

— Structure of the document

This report contains a list of issues and comments on all the above source files. Each issue is assigned a severity level based on the potential impact of the issue and recommendations to fix it, if applicable. For ease of navigation, an index by topic and another by severity are both provided at the beginning of the report.

— Documentation

For this audit, we used the following sources of truth about how the swap functions should work:

https://ftxclassic.net

These were considered the specification.

— Comments from Auditor

No vulnerabilities with critical, high, medium or low-severity were found in the above source code.

Additional notice: 0.

02. List of issues by severity

List of issues by severity

A. Critical
- N/A

B. High
- N/A

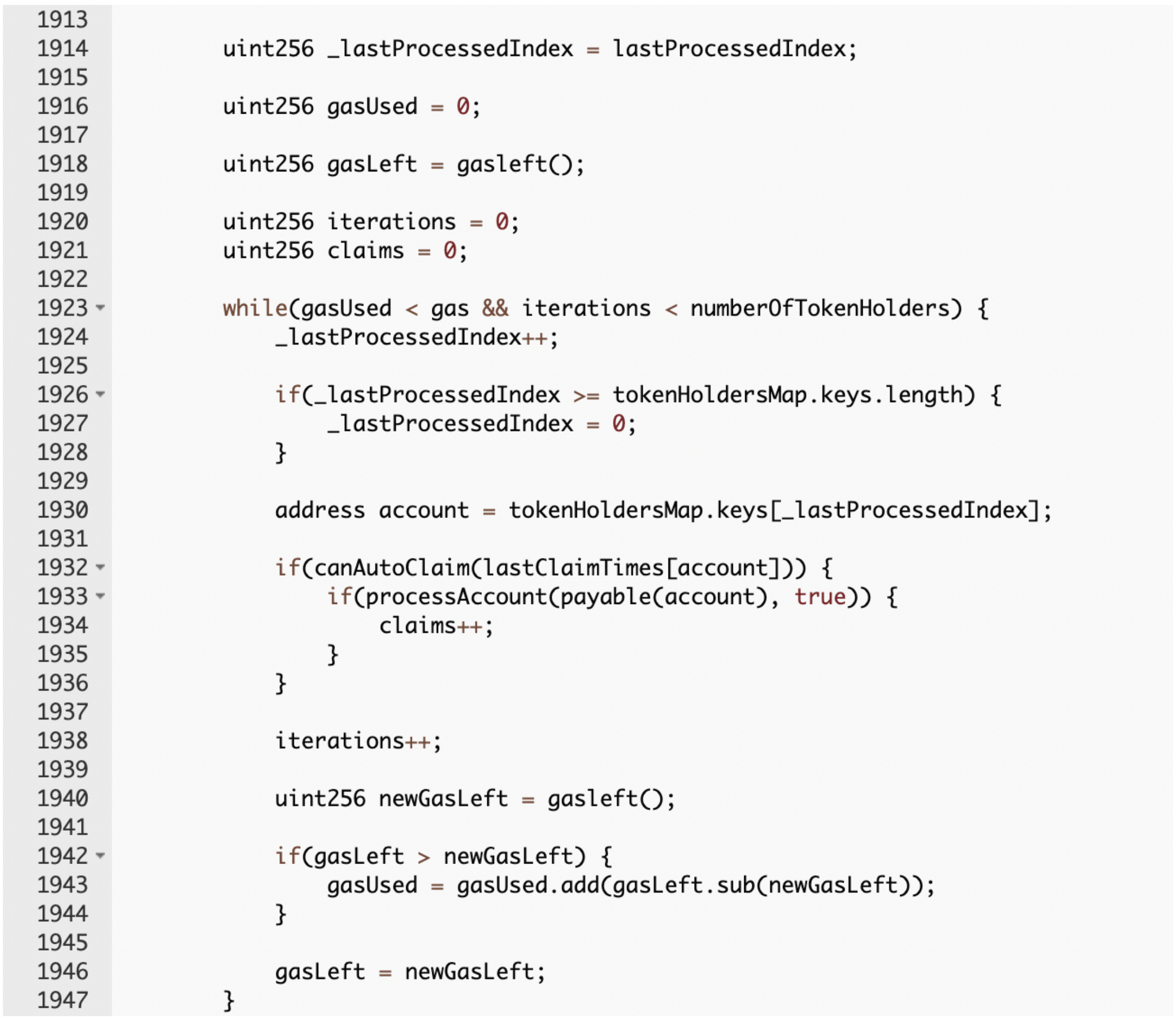
C. Medium
- N/A

D. Low
- N/A

E. Informational

EEH-02 | Variable Declare as Constant

Category	Severity	Location	Status
Gas Optimization	● Informational	1914 - 1947	ⓘ Acknowledged



```
1913
1914         uint256 _lastProcessedIndex = lastProcessedIndex;
1915
1916         uint256 gasUsed = 0;
1917
1918         uint256 gasLeft = gasleft();
1919
1920         uint256 iterations = 0;
1921         uint256 claims = 0;
1922
1923         while(gasUsed < gas && iterations < numberOfTokenHolders) {
1924             _lastProcessedIndex++;
1925
1926             if(_lastProcessedIndex >= tokenHoldersMap.keys.length) {
1927                 _lastProcessedIndex = 0;
1928             }
1929
1930             address account = tokenHoldersMap.keys[_lastProcessedIndex];
1931
1932             if(canAutoClaim(lastClaimTimes[account])) {
1933                 if(processAccount(payable(account), true)) {
1934                     claims++;
1935                 }
1936             }
1937
1938             iterations++;
1939
1940             uint256 newGasLeft = gasleft();
1941
1942             if(gasLeft > newGasLeft) {
1943                 gasUsed = gasUsed.add(gasLeft.sub(newGasLeft));
1944             }
1945
1946             gasLeft = newGasLeft;
1947         }
```